Lecture two

Saturday, March 20, 2010
12:51 PM

Examples of Decidable languages :-

- $A_{DFA} = \{\langle B, w\rangle :$ B is a DFA that accepts input string $w\}$ is decidable.

TM M that decides A_{DFA}

M = " On input $\langle B, w\rangle$, where B is a DFA and w is a string:

1. Simulate B on input w.
2. If the simulation ends in an accept state, accept. If it ends in a non-accepting state, reject ."

- $A_{NFA} = \{\langle B, w\rangle :$ B is an NFA that accepts string $w\}$. is decidable.

N1 = " On input $\langle B, w\rangle$:

1. Simulate NFA B on input w.
2. If simulation ends on an accepting state, accept. If it ends in non-accepting state, reject ."

N1 is a non-deterministic Turing machine.

N2 = " On input $\langle B, w\rangle$:

1) Convert NFA B to an equivalent DFA C.
2) Run TM M (from previous example) on input $\langle C, w\rangle$.
3) Accept if M accepts, reject if M rejects ."

N2 is deterministic TM.

- $A_{REX} = \{\langle R, w\rangle :$ R is a regular expression and R generates $w\}$ is decidable.

P = " On input $\langle R, w\rangle$:

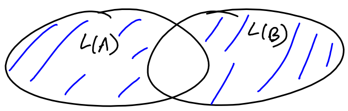
1. Convert R into NFA A.
2. Run TM N2 on $\langle A, w\rangle$.
3. If N2 accept, accept, if N2 rejects, reject."

- $E_{DFA} = \{ \langle A \rangle : A$ is a DFA and $L(A) = \phi \}$ is decidable.

$T =$ " On input $\langle A \rangle$:

1. Mark the start state of A.
2. Repeat until no new states are marked :
   a) Mark any state that has a transition coming into it from any state that is already marked.
3. If no accept state is marked, accept ; otherwise reject ."

- $EQ_{DFA} = \{ \langle A, B \rangle : A$ and B are DFA and $L(A) = L(B) \}$.



Blue : Symmetric Difference of $L(A)$ and $L(B)$.

$$L_1 = \left(L(A) \cap \overline{L(B)} \right) \cup \left(\overline{L(A)} \cap L(B) \right)$$

We know from properties of regular languages that L_1 is regular. Consider a DFA C for L_1.

$F =$ " On input $\langle A, B \rangle$:

1. Construct DFA C as described.
2. Run TM T from previous example on input $\langle C \rangle$.
3. If T accepts, accept ; if T rejects, reject ."

- $A_{CFG} = \{ \langle G, w \rangle : G$ is a CFG that generates string $w \}$ is decidable.

$S =$ " On input $\langle G, w \rangle$:

1. Convert G to an equivalent grammer in Chomsky normal form.
2. List all derivations with $2n-1$ steps, where $n = |w|$. (if $n=0$, list all derivations of length 1).
3. If any of these derivations accept w, accept ; if not reject."

Fact :- If grammer G is in Chomsky normal form, for any string w ($n = |w|$), any derivation of w has $2n-1$ steps

- $E_{CFG} = \{ \langle G \rangle : G$ is a CFG and $L(G) = \phi \}$ is decidable.

R = " On input $\langle G \rangle$:

1. Mark all terminal symbols in G.
2. Repeat until no new variables get marked:

   a) Mark any variable A where G has a rule

   $A \to u_1 u_2 \dots u_k$ and

   each symbol $u_1, \dots, u_k$ has already been marked.

3. If the start variable is not marked, accept; otherwise reject."

Idea:- Determine for each variable, inparticular also for the start variable, if it is capable of generating a string of terminals.

- Every context free language A is decidable.

  Let G be a context free grammer for A.

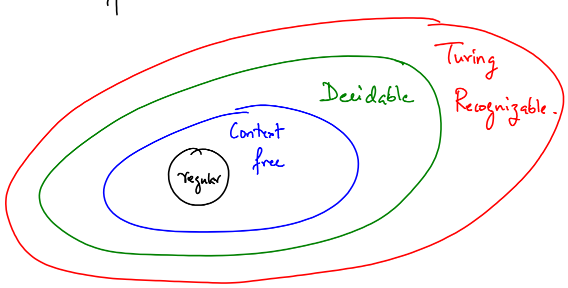
  M_G = " On input w:

  1. Run TM S (as defined previously) on input $\langle G, w \rangle$.
  2. If S accept, accept; if S rejects, reject. "

- $EQ_{CFG} = \{ \langle G, H \rangle$: G and H are CFGs and $L(G) = L(H) \}$.

  Cannot use arguments as for Regular expressions since context free languages are not closed under complementation and intersection.

Infact EQ_{CFG} is NOT decidable.



Undecidable Languages:

$A_{TM} = \{ \langle M, w \rangle$: M is a TM and M accepts w$\}$.

A_{TM} is recognizable.

U = " On input $\langle M, w \rangle$:

1) Run M on input w.

2) Accept if M accepts; Reject if M rejects. "

2) Accept if M accepts. Reject if M rejects."

U is called the 'Universal Turing Machine'.

We will show A_{TM} is NOT decidable.

The Diagonalization Method :-

A function $f: A \to B$ is one-to-one if

$\forall a_1, a_2 \in A, \quad a_1 \neq a_2 \qquad f(a_1) \neq f(a_2)$

A function $f: A \to B$ is onto if

$\forall b \in B, \ \exists a \in A \ \text{ s.t. } \ f(a) = b.$

Function f is called a 'Correspondence' if it is one-to-one and onto.

George Cantor (1873) : Two sets A and B are of same size iff there exists correspondence $f: A \to B$.

Examples:- • $\{1, 2\}$ same size $\{5, 8\}$.

- Set of natural numbers $\mathcal{N} = \{1, 2, 3, \ldots\}$ same size as set of even numbers $\mathcal{E} = \{2, 4, 6, 8 \ldots\}$.
- Set of rational numbers $\mathbb{Q} = \left\{ \frac{m}{n} : m, n \in \mathcal{N} \right\}$ same size as $\mathcal{N}$.

$$
\begin{matrix}
\frac{1}{1} & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \cdots\cdots \\
\frac{2}{1} & \frac{2}{2} & \frac{2}{3} & \frac{2}{4} & \frac{2}{5} & \cdots\cdots \\
\frac{3}{1} & \frac{3}{2} & \frac{3}{3} & \frac{3}{4} & \frac{3}{5} & \cdots\cdots \\
\frac{4}{1} & \frac{4}{2} & \frac{4}{3} & \frac{4}{4} & \frac{4}{5} & \cdots\cdots
\end{matrix}
$$

Defn : (Countable Set) : A set is countable if it is finite or has the same size as $\mathcal{N}$.

A set which is not countable is called uncountable.

Thm :- The set of real numbers $\mathbb{R}$ is uncountable.

Proof:- Assume for contradiction that $\mathbb{R}$ is countable. Let $f: \mathcal{N} \to \mathbb{R}$ be a correspondence between $\mathcal{N}$ and $\mathbb{R}$.

Let $x = 0.x_1 x_2 x_3 \ldots\ldots$

such that $x_i \neq \{ i\text{th digit of } f(i), 0, 9 \}$. Then $x \notin$ Range of f.

Hence f is not onto and hence contradiction.

Thm:- Some (allmost all) languages are not recognizable.

Proof:- The set of all Turing machines is countable.

The set of all languages is uncountable.

Thm:- $A_{TM} = \{ \langle M, w \rangle : M$ is a TM and M accept $w \}$ is undecidable.

Proof:- Assume for contradiction that A_{TM} is decidable. Let H be a decider for A_{TM}.

$$H(\langle M, w \rangle) = \begin{cases} \text{accept} & \text{if } M \text{ accepts } w \\ \text{reject} & \text{otherwise.} \end{cases}$$

Let $D =$ " On input $\langle M \rangle$:

1. Run H on $\langle M, \langle M \rangle \rangle$.
2. Accept if H rejects; reject if H accepts. "

$$D(\langle M \rangle) = \begin{cases} \text{reject} & \text{if } M \text{ accepts } \langle M \rangle \\ \text{accept} & \text{if } M \text{ does not accept } \langle M \rangle. \end{cases}$$

$$D(\langle D \rangle) = \begin{cases} \text{reject} & \text{if } D \text{ accepts } \langle D \rangle. \\ \text{accept} & \text{if } D \text{ does not accept } \langle D \rangle. \end{cases}$$

Contradiction! $\square$

Definition (co-Turing-recognizable): Language A is co-Turing-recognizable if $\overline{A}$ is Turing-recognizable.

Thm: A language B is decidable iff B is Turing-recognizable and co-Turing-recognizable.

Proof:- ($\Rightarrow$) Easy to argue

($\Leftarrow$) Let M_1 be recognizer of B and M_2 be recognizer of $\overline{B}$.

$M =$ " On input w:

1. Run both M_1 and M_2 on input w in parallel.
2. If M_1 accept, accept; if M_2 accept, reject." $\square$

Corollary: $\overline{A_{TM}}$ is NOT Turing-recognizable.

Proof:- We know A_{TM} is Turing-recognizable, and not decidable. $\square$.